# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CSC174 server-side javascript**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Unit 04 LAB: rabbit population forecast

# Objectives

In this lab assignment, students will learn:

* Why Node.js is not the best option for CPU intensive tasks
* The difference between multi-threaded and single threaded web servers
* How Node.js handles HTTP requests in a single threaded environment
* How to use built-in console time() and timeEnd() functions to track time
* What thread starvation is

# COURSE PREPARATION

You should have done your reading assignment listed under “Reading Assignment” and “Video Assignment” sections in BlackBoard. You should also have reviewed the lecture slides in BlackBoard. There is an optional section called “In Case You Don’t Know” in BlackBoard for those who have limited exposure to JavaScript language.

# WHat to submit

For this lab you need to submit the following files:

* **rabbit.js (Part One)**
* **Short answer questions sheet (Part Two)**

# grading rubric:

Be sure to follow the Coding Standard Guidelines. You must properly indent and comment your code. This assignment is worth 100 points. (70 points from Part One and 30 points from Part Two)

* Indent code and insert comments to document your program. [5 pts]
* Program must be implemented and run with no syntax errors. [30 pts]
* Program must be implemented and run with no logic errors. [30 pts]
* Required source files should be zipped and uploaded to BlackBoard assignment drop box before the deadline. [5 points]

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Part One: Rabbit Population Forecast**

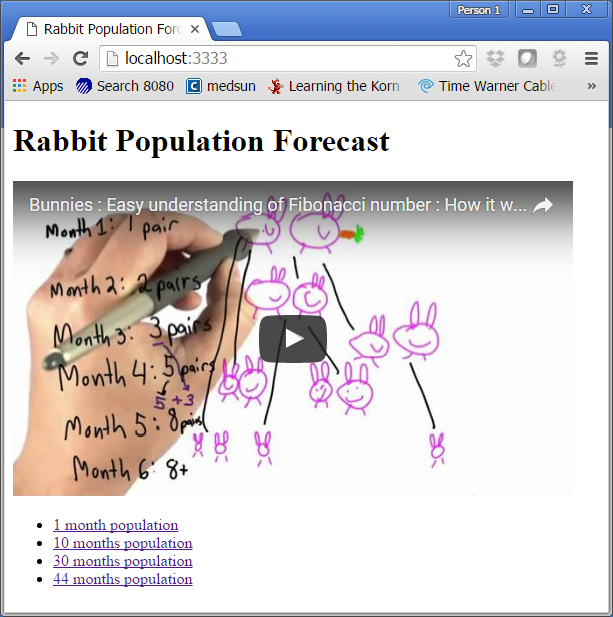
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Project Description:**

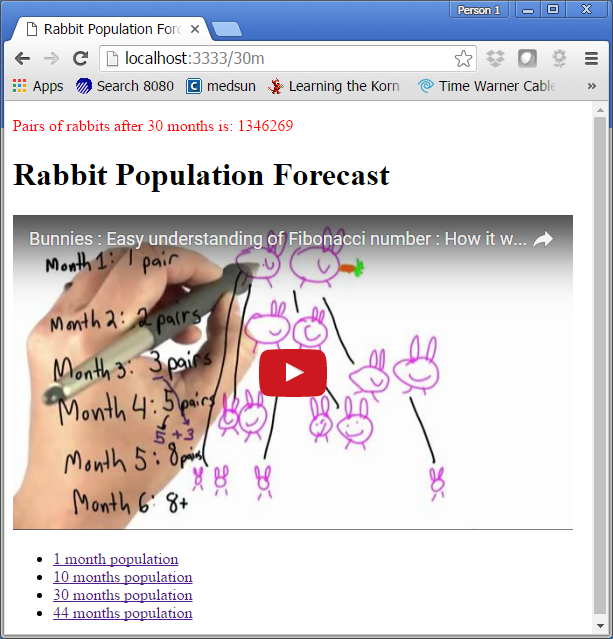
You are given a Node.js project that calculates rabbit population in 1, 10, 30, and 44 months respectively. This project is adapted from the example in the textbook Listing 2-24 on page 29. Your job is to modify this Node.js program (rabbit.js) by adding another entry for 55 months population.

The instructor should provide you the source code in ***unit04\_lab.zip*** file in BlackBoard Unit 04 Lab sections. There is only one file in this folder: ***rabbit.js***.

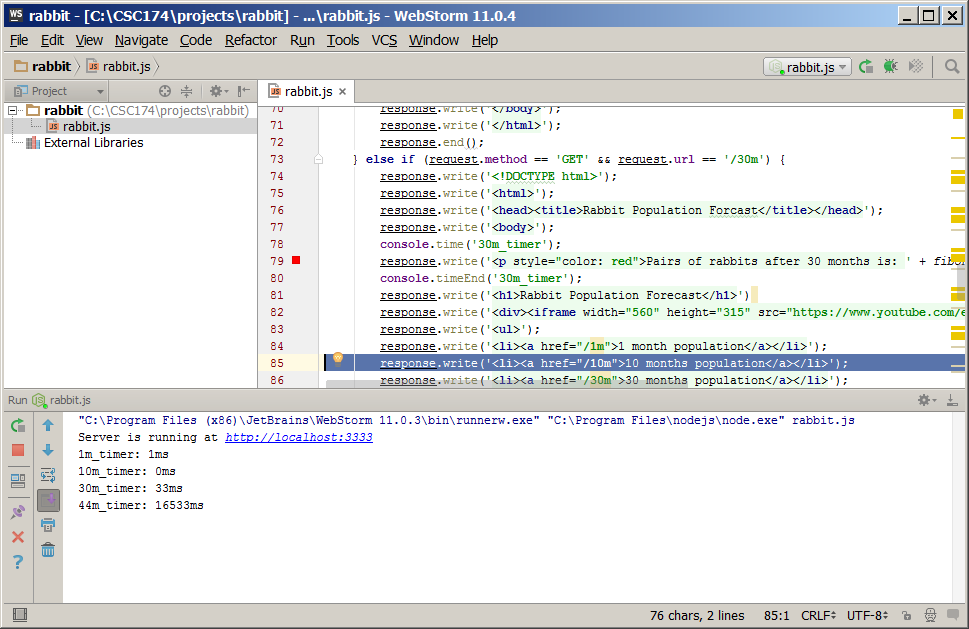
When you run this project, you should see something like this in the browser:



If you click any links at the bottom, you should get something like this:



In the meantime your Node.js server log should give you the number of milliseconds it takes to calculate this number. See screenshot below.



As you can see from the timers, it takes significantly longer to calculate for 44 months. To make the matter worse, when the Node.js is calculating for 44 months’ population, the whole application gets stuck. This shows that Node.js uses a single thread to process multiple requests, so if one request causes a lot of CPU consumption in Node.js server, other requests have to wait.

Now add another entry for 55 months population, then try to click on it, your Node.js may hang on you and you may never get a response back. This is thread starvation. You have to stop the Node.js server and restart to recover.

You only need to submit the modified rabbit.js file.

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Part Two: Short answer questions**

\*All questions are based on the original rabbit.js source code provided by the instructor.

Question 1: What port is this application running on?

3333

Question 2: How many function definitions do you see in this program? List the function names. If it’s an anonymous function write the function definition.

2

Fibbonaci(n)

Var server = http.createServer(function (request, response) {

If and if else statements for the webpages

}

Question 3: How many global variables are there in this program? What are they? (Hint: Some are programmer defined using keyword ***var*** and some provided by JavaScript language implicitly.)

3

http

server

console